// assets/js/app.js

document.addEventListener('DOMContentLoaded', function() {

```
  // --- НАДЕЖНОЕ УПРАВЛЕНИЕ МОДАЛКОЙ С ДЕБАГ-МАЯКОМ ---
 // --- УПРАВЛЕНИЕ ЗАГРУЗЧИКОМ ПРОЕКТОВ ---
  const btnCreateProject   = document.getElementById('btn-create-project');
  const projectModal       = document.getElementById('project-modal');
  const btnModalCancel     = document.getElementById('btn-modal-cancel');
  const btnModalSubmit     = document.getElementById('btn-modal-submit');
  const newProjectName     = document.getElementById('new-project-name');
  const modalErrorBox      = document.getElementById('modal-error-box');
  const folderInput        = document.getElementById('project-folder-
input');
  const btnSelectFolder    = document.getElementById('btn-select-folder');
  const folderStatus       = document.getElementById('selected-folder-
status');
```

```
  if (btnCreateProject && projectModal) {
   // Функция проверки условий для активации кнопки "Загрузить"
      function checkInputsValidity() {
          const nameVal = newProjectName.value.trim();
          const hasFiles = folderInput.files && folderInput.files.length >
0;
          if (nameVal !== "" && hasFiles) {
              btnModalSubmit.disabled = false; // Включаем кнопку
              btnModalSubmit.style.opacity = "1";
          } else {
              btnModalSubmit.disabled = true;  // Выключаем кнопку
              btnModalSubmit.style.opacity = "0.6";
          }
      }
```

```
   // Следим за вводом имени
      newProjectName.addEventListener('input', checkInputsValidity);
```

```
   // Клик по кастомной кнопке -> триггерим скрытый системный инпут выбора папки
      btnSelectFolder.addEventListener('click', function() {
          folderInput.click();
      });
```

```
   // Следим за тем, что пользователь выбрал папку на ПК
      folderInput.addEventListener('change', function() {
          if (this.files && this.files.length > 0) {
       // Берем имя корневой папки, которую выбрал юзер
              const firstFile = this.files[0];
              const rootFolderName =
firstFile.webkitRelativePath.split('/')[0];
              folderStatus.innerHTML = `Выбрано файлов:
<strong>${this.files.length}</strong><br><span style="font-size:11px;
color:#2da44e;">(из папки: ${rootFolderName})</span>`;
```

```
            } else {
                folderStatus.textContent = "Папка не выбрана";
            }
            checkInputsValidity();
        });
```

```
    // Открытие модалки
        btnCreateProject.addEventListener('click', function() {
            newProjectName.value = '';
            folderInput.value = ''; // Сбрасываем выбранные файлы
            folderStatus.textContent = "Папка не выбрана";
            if (modalErrorBox) modalErrorBox.classList.add('hidden');
            btnModalSubmit.disabled = true;
            btnModalSubmit.style.opacity = "0.6";
            projectModal.classList.remove('hidden');
            newProjectName.focus();
        });
```

```
    // Отмена
        btnModalCancel.addEventListener('click', function() {
            projectModal.classList.add('hidden');
        });
```

```
    // Нажатие на кнопку "Загрузить" -> отправка терабайтов кода по AJAX
        btnModalSubmit.addEventListener('click', async function() {
            const projectName = newProjectName.value.trim();
            if (modalErrorBox) modalErrorBox.classList.add('hidden');
```

```
            const formData = new FormData();
            formData.append('name', projectName);
```

```
      // КРИТИЧЕСКИ ВАЖНО: Пакуем файлы вместе с их путями вложения
            for (let i = 0; i < folderInput.files.length; i++) {
                let file = folderInput.files[i];
                formData.append('project_files[]', file);
        // Передаем бэкенду относительный путь (например: "my_proj/src/main.php")
                formData.append('project_paths[]', file.webkitRelativePath);
            }
```

```
      // Меняем статус кнопки на время тяжелой загрузки
            btnModalSubmit.disabled = true;
            btnModalSubmit.textContent = "Загрузка...";
```

```
            try {
                const response = await
fetch('backend/router.php?action=create_project', {
                    method: 'POST',
                    body: formData
                });
                const result = await response.json();
```

```
            if (result.success) {
                projectModal.classList.add('hidden');
                btnModalSubmit.textContent = "Загрузить";
                location.reload();
            } else {
                btnModalSubmit.disabled = false;
                btnModalSubmit.textContent = "Загрузить";
                if (modalErrorBox) {
                    modalErrorBox.textContent = result.error;
                    modalErrorBox.classList.remove('hidden');
                }
            }
        } catch (error) {
       // Возвращаем кнопку в исходное рабочее состояние
            btnModalSubmit.disabled = false;
            btnModalSubmit.textContent = "Загрузить";
       // КРАСИВО ВЫВОДИМ ОШИБКУ СЕТИ В ТВОЮ ПЛАШКУ БЕЗ ВСЯКИХ АЛЕРТОВ
            if (modalErrorBox) {
                modalErrorBox.textContent = "Критическая ошибка отправки
файлов на сервер.";
                modalErrorBox.classList.remove('hidden');
            }
        }
    });
  }
```

```
 // Локальное хранилище для текстов текущего выбранного релиза,
 // чтобы переключать вкладки мгновенно без повторных запросов к серверу
  let currentReleaseTexts = {
      readme: '',
      license: '',
      comment: '',
      log: 'Здесь будет выводиться лог действий (100 строк)...'
  };
```

```
  /**
 * Отрисовка таблицы файлов в верхнем окне
   * @param {Array} files Массив файлов из JSON
   */
  function renderFilesTable(files) {
      if (!files || files.length === 0) {
          filesList.innerHTML = `<tr><td colspan="4" class="text-center
text-muted">В данном релизе нет файлов для скачивания.</td></tr>`;
          return;
      }
```

```
      let html = '';
      files.forEach(file => {
          const icon = file.is_dir ? '' : '';
          html += `
              <tr>
```

```
                    <td><span style="margin-right: 8px;">${icon}</span>
${escapeHtml(file.name)}</td>
                    <td class="text-muted">—</td> <!-- Примечания (коммиты)
прикрутим позже через .meta.json -->
                    <td class="text-muted">${file.date}</td>
                    <td style="text-align: right;"><span class="text-muted"
style="font-size: 12px;">${file.size}</span></td>
                </tr>
            `;
        });
        filesList.innerHTML = html;
    }
```

```
 // Вспомогательная функция защиты от XSS при отрисовке имен файлов
    function escapeHtml(string) {
        return String(string).replace(/&/g, '&amp;').replace(/</g,
'&lt;').replace(/>/g, '&gt;').replace(/"/g, '&quot;');
    }
```

```
 // 1. СЛУШАЕМ ВЫБОР РЕЛИЗА В ВЫПАДАЮЩЕМ СПИСКЕ
    releasesDropdown.addEventListener('change', async function() {
        const selectedRelease = this.value;
```

```
        if (!selectedRelease) {
       // Если сбросили выбор — возвращаем интерфейс в исходное состояние
            btnDownloadZip.disabled = true;
            filesList.innerHTML = `<tr><td colspan="4" class="text-center
text-muted">Выберите релиз из списка выше для просмотра файлов...</td></tr>`;
            textEditor.value = 'Текст файла отсутствует или релиз не выбран...';
            return;
        }
```

```
        filesList.innerHTML = `<tr><td colspan="4" class="text-center text-
muted">Загрузка данных релиза...</td></tr>`;
```

```
     // Дергаем наш апи-модуль
        const result = await Api.getReleaseData(selectedRelease);
```

```
        if (result.success) {
       // Активируем кнопку ZIP
            btnDownloadZip.disabled = false;
       // Сохраняем тексты в память для вкладок
            currentReleaseTexts.readme = result.texts.readme || 'Файл
README.txt пуст.';
            currentReleaseTexts.license = result.texts.license || 'Файл
LICENSE.txt пуст.';
            currentReleaseTexts.comment = result.texts.comment || 'Файл
COMMENT.txt пуст.';
       // Отрисовываем файлы в таблице верхнего окна
            renderFilesTable(result.files);
```

```
    // Автоматически показываем текст той вкладки, которая сейчас активна (по умолчанию README)
          const activeTab = document.querySelector('.tab-btn.active').getAttribute('data-target');
          textEditor.value = currentReleaseTexts[activeTab];
      } else {
          alert("Ошибка загрузки: " + result.error);
          filesList.innerHTML = `<tr><td colspan="4" class="text-center" style="color: #cf222e;">${result.error}</td></tr>`;
      }
  });
```

```
 // 2. СЛУШАЕМ ПЕРЕКЛЮЧЕНИЕ ВКЛАДОК НИЖНЕГО ОКНА
  tabButtons.forEach(button => {
      button.addEventListener('click', function() {
          tabButtons.forEach(btn => btn.classList.remove('active'));
          this.classList.add('active');
          const targetTab = this.getAttribute('data-target');
     // Вытаскиваем текст из памяти без лишних запросов к  PHP
          textEditor.value = currentReleaseTexts[targetTab] || `Текст для вкладки ${targetTab.toUpperCase()} отсутствует.`;
      });
  });
```

```
});
```

From:
https://wwoss.ru/ - **worldwide open-source software**

Permanent link:
https://wwoss.ru/doku.php?id=tmp_27.06.2026

Last update: **2026/06/27 15:35**